MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE
NATIONAL TECHNICAL UNIVERSITY
“KHARKIV POLYTECHNIC INSTITUTE”

Guidance for course work on “Algorithms and Data Structures”

Методичні вказівки до виконання курсової роботи по курсу «Алгоритми та структури даних»

for students of direction
6.050103“Software Engineering”, specialty .05010302 “Software Engineering”

для студентів, які навчаються за напрямком
6.050103 «Програмна інженерія» спеціальності .05010302«Інженерія програмного забезпечення»

Затверджено
редакційно-видавничою радою
НТУ «ХПІ», протокол № 2
від «23» червня 2016р.

Kharkiv
NTU“KPI”
2016

Guidance for course work on “Algorithms and Data Structures” for students of for students of direction 6.050103“Software Engineering”, specialty .05010302 “Software Engineering” / comp. Stratiienko N.K., Shmatko O.V., Borodina I.O. – Kh.: NTU“KPI”, 2016. – 30 p.

Compilers: N.K. Stratiienko,
O.V. Shmatko,
I.O. Borodina

Reviewer V.O. Guzhva

Software Engineering and Management Information Technologies Department

TABLE OF CONTENTS

INTRODUCTION

Guidance for course work on “Algorithms and Data Structures” is a part of methods of the course “Algorithms and Data Structures”.

Course work is performed in 3 semester.

The course work must consolidates theoretical knowledge and practical skills obtained in the study of lecture and practical parts of the course.

The guidance covers the key issues related to the course work, the processing of explanatory notes to course work, and the defense of student work.

1 OBJECTIVE AND TASKS OF WRITING A COURSE WORK

Course work is carried out to consolidate the knowledge of the course “Algorithms and Data Structures” and get software development skills using studied data structures and algorithms.

The objectives of the course are:

– acquire skills in developing programs that include non-trivial algorithms;

– develop of algorithmic software;

– develop software.

The results of the course work include developed software, explanatory note to the course work, and presentation material.

Explanatory note to the course work is the basic document that reflects all the stages and results of the performed work. An explanatory note must be written according to rules from regulatory guidance publication of NTU “KhPI” “Texts in the educational process’s Requirements to fulfill" (НТУ “ХПІ” “Текстові документи у сфері учбового процесу. Основні вимоги до виконання”).

An explanatory note should contain:

– task for course work;

– supervisor's review;

– abstracts in Ukrainian, Russian, and English;

– table of contents;

– introduction;

– main part;

– conclusion;

– list of sources of information;

– appendices.

2 MAIN STAGES OF PERFORMING A COURSE WORK

Term paper for the course “Algorithms and Data Structures” requires development of algorithms and software according to the resulting problem.

The course work is divided into the following stages:

– research problem statement;

– construction of mathematical models;

– study of the theoretical foundations of algorithms and in-depth study of data structures to be used;

– development and testing of algorithms and software;

– solve problems and analyze results;

– registration explanatory notes and presentations;

– defense of student work.

All steps must be completed within time that is specified in the schedule of implementation.

The research is divided into three stages.

At the preparatory stage, each student must:

– be acquainted with the received problem;

– explore the algorithms and data structures required to accomplish the task;

– develop algorithms and software;

– construct a set of test data.

At the working stage of the study, the student must:

– solve the problem for obtained test data using the developed software to verify the written algorithms;

– solve the problem for high dimensional data to identify the limits of use of the software.

At the final stage of the study, the student must:

– interpret the results in terms of object domain of the task;

– draw conclusions on the results of the work.

2.1 Research problem statement

Research problem statement is the most important stage of operational research since errors at this stage can make useless all the subsequent stages. First, task is defined in terms of the customer. After determining the purpose, the object is investigated, and a set of factors affecting the process flow is defined. Then the researcher identifies a set of significant factors in consultation with the customer and finally refines quality (content) statement of the problem.

There is a simplified situation in this course work: students have to use problem statement from their tasks for course work, but they may contact the supervisor for additional information.

2.2 Construction of the mathematical model

Having a rigorous, logically consistent meaningful statement of the problem, student performs its formalization by constructing a mathematical model that contains the corresponding optimality criteria and restrictions. Apparatus for constructing such a model is studied in the lecture course.

2.3 Exploring the theoretical foundations of the used methods

Having a mathematical model, student classifies tasks and algorithms to apply, takes an overview of key algorithms of this class, study in detail the algorithms and data structures listed in the task.

2.4 Development and testing of algorithms and software

The software must be friendly to the researcher and use principles of modern information technologies.

Requirements to the software must be described by a use case diagram.

Many variants of tasks in this course work require a step by step execution and graphical interpretation. Additionally, one needs to generate a report on the results of the program in the form of plain text, HTML, LaTeX, or PDF. In addition, user should be able to get help on the program.

According to the task for course work, the main window should include (besides menu) view and edit controls, buttons to perform some functions and controls to display results. It is advisable to place panels for graphical representation of results.

Many task variants require reading data from a large binary file. It is also recommended to add the ability to store and read data in text format.

There should be a small dialog box that contains the name of the program and information about its author.

Software testing is performed on problems of small dimension that can be solved manually and compared with the calculations of the program.

3 COURSE WORK CONTENT

This chapter discusses an example of structure of course work report. A report consists of introduction, four chapters, conclusion, and appendices. Appendices can contain illustrations, tables, and formulas. Minimum size of a report is 40 pages.

A course work report can be accompanied by a storage containing full information on all course work artifacts:

– source code of the developed software;

– executable files, such as ELF format or JAR; Portable Executable (. exe) is not recommended;

– full text of the report.

We will give a brief description of contents of sections and subsections of a report. This description is just a recommendation. It reflects the necessary information that must be given in a report. Student can change report structure and content after consultation with his supervisor.

3.1 Introduction

An introduction discusses the goals that must be achieved as a result of the course work. Course work actuality must be proved. Estimated size is 1-1.5pages.

3.2 Chapter 1 THEORETICAL FOUNDATIONS OF ALGORITHMS DESIGN AND ANALYSIS. PROBLEM STATEMENT

The main purpose of this section is to show that the student is familiar with the basics of the theory of algorithms and required data structures, and to formulate basic requirements for software development.

Section 1.1 Mathematical foundations of algorithms analysis

The section should define the following basic concepts of the theory of algorithms: algorithm complexity, "divide-and-conquer" algorithms, dynamic programming, and amortized analysis.

Estimated size is 3-4 pages.

Section 1.2 Data structures and classification of basic algorithms

The section should give a description of the data structures and classification of basic algorithms: graph algorithms, dynamic programming, geometric algorithms, and so on.

Estimated size is 3-4 pages

Section 1.3 Problem statement

Student must:

– give a mathematical model of the problem;

– bring qualitative statement of the problem;

– describe the main objectives to be achieved as a result of the course work.

The main tasks of the course work are:

– choice of platform (operating system) for the software;

– choice of software development tools;

– development of software;

– debugging and testing of software;

– trial operation of software on control data and results analysis.

Estimated size is 6-8 pages.

3.3 Chapter 2 THEORETICAL FOUNDATIONS OF SOFTWARE ENGINEERING

The chapter is devoted to mathematical software for solving the problem.

Section 2.1 Data structures and classification of basic algorithms of given class

The section should describe a given class of algorithms and data structures, for example, describe the graph algorithms and how to represent a graph in memory.

Estimated size is 2-3 pages.

Section 2.2 Description of the algorithms

The section should describe specific algorithms that will be used in the work. It should give the algorithm schemes in the form of a well-structured

pseudocode or real code. Show how to determine complexity of the described algorithms and provide examples of running the algorithms on simple data.

Estimated size is 6-9 pages.

3.4 Chapter 3 DESCRIPTION OF THE DEVELOPED SOFTWARE

The main purpose of this chapter is to describe the results obtained during software development.

Section 3.1 Definition of software use cases

The section should identify the main types of users working with the software to describe use cases. Use case diagram must be present.

Examples of possible use case diagram are given in Appendix B.

Estimated size is 2-3 pages.

Section 3.2 Rationale for operating system and software development tools

This section should describe features of 2-3 operating systems and 2-3 software development tools. Main advantages and disadvantages of these software products must be given.

The section should ground the choice of operating system and software development tools.

Estimated size is 3-6 pages.

Section 3.3 Structure of the application

This section should describe the key components of the developed software and relationships between them. It is necessary to introduce UML component and classes diagrams and describe them in text.

Diagram shows the various components of the system and relations between them. A component is a physical module of code. Component is often considered as a synonym to package, but these concepts may differ because components are physical associations of code. Although a separate class can be represented in a set of components, this class must be defined in just one package. For example, String class in Java is part of the java.lang, but it can be in a number of components.

Dependencies between components show how changes in one component can affect changes in other components. There is a very limited number of types of dependencies that can be used, including "link" and "compile" dependencies.

Class diagram describes the types of objects and different kinds of static relationships that exist between them. There are two basic kinds of static relationships:

– associations (e.g., customer can rent a number of video tapes);

– subtypes (a nurse is a kind of person).

The diagrams can contain attributes of classes, operations, and constraints imposed on relationships between objects.

Estimated size is 2-5 pages.

3.5 Chapter 4 USING THE DEVELOPED SOFTWARE

The main purpose of this chapter is to describe the results obtained through use of the developed software to solve applied problems. This section describes how user works with the software. Also, it describes and analyzes the results obtained during solving practical problems.

Section 4.1 Installing the software

This section describes how to install the developed software on user's computer. Installation procedure should be described as a sequence of steps performed by user during installation. It is also recommended to bring the basic requirements for hardware and software: the type of operating system, presence of special software, processor type, the minimum amount of RAM, amount of free space on hard drive, and so on.

If student selects GNU / Linux operating system, it is advised to package the software (rpm, deb, or ebuild) because it greatly simplifies installation of software dependencies and removing it from the system.

Sequence of actions during the first start of the application must be described (i.e., how to start the application and how exit from it).

Estimated size is 1-2 pages.

Section 4.2 User manual

This section describes how user should work with the developed software.

This description can be treated as a part of user guide that comes with documentation for the software. The description should be sufficiently detailed to allow work for user that has no special training with the developed software. Description must be illustrated with screenshots.

Estimated size is 6-10 pages.

Section 4.3 Analysis of results

This section describes the results obtained when using the application software.

All large dimension input data or their fragment must be shown in form of tables or figures. Screenshots can be used for that.

The section must contain results and analyze them.

Estimated size is 2-4 pages.

3.6 Conclusion

Conclusion summarizes the course work, lists its results shortly, describes performance of the developed application and its feasibility for solving practical problems.

Size of the chapter is 1 page.

3.7 List of sources

The chapter lists cited, discussed, and used sources of information. Sources of information are books, articles, legal and technical papers, dissertations, and so on. Sources of information go in single list without respect if they were or were not referenced in the text.

The sources referred in the text are placed in the order of their appearance.

Sources without references are placed as follows:

1) source edited by one of the authors are listed in alphabetical order of their titles;

2) books, articles, reviews, abstracts go in alphabetical order of author names;

3) normative and technical documents are grouped into categories and within categories by their registration numbers.

4 REQUIREMENTS TO DEVELOPED SOFTWARE

4.1 Choice of software development tools

Choice of software development tools must be done taking into account the following considerations:

– experience in working with this software;

– features of this software, in particular, the availability of licenses;

– efficiency of its use in the application software development ;

– rapid visual design of application software;

– convenient debug;

– getting advices.

Tasks can be perform in a chosen programming language:

– C;

– C++;

– Pascal (rarely used today);

– Java;

– C#;

– Python;

– Ruby.

Assembly language should be avoided because they add too many details to the algorithm.

The program can be demonstrated on any platform. It is recommended to develop an application so that it could be built and run on a wide class of systems (it is easy to reach it using the recommended languages). It is recommended to use any GNU / Linux distro or FreeBSD.

Choosing a development environment may be dictated by the student's habits. It is advisable to use multiple environments for different works.

Here are a few important free/open editors and development environments:

– Vim;

– Emacs;

– Eclipse;

– NetBeans;

– IntelliJ IDEA;

– KDevelop;

– MonoDevelop;

– Lazarus (a free environment like Delphi).

4.2 Software structure

The developed application must implement the following functions:

1) enter, edit, and save initial data;

2) solve the basic problem;

3) verify the solution;

4) view solution search process step by step;

5) save a report;

6) get help.

The following section describes composition and characteristics of each program.

4.2.1 Enter, edit, and save the initial data

It is necessary to provide entering initial data manually or from a file, edit data, and save them to a file. The exact file format is chosen by the student. We recommend using text format (plain text, XML, JSON).

Many task variants require reading data from a large binary file. It is not necessary to save the data back in binary format, but saving to the basic format (text) must be implemented.

4.2.2 Solve the basic problem

It is necessary to solve the problem using given algorithms. If conditions are incompatible or input format is incorrect, application should notify the user. The program should work so that it finds the solution or reports an error for any input.

In addition, the application must provide the ability to view the process of finding solution step by step, even if the task has incompatible condition that can allow several steps of the algorithm.

4.2.3 Verify the solution

Many task variants allow verification of the solution.

Testing can be performed using a graphical interpretation, for example, elements of the sorted sequence can be represented as a histogram.

Tasks such as the generation of random sequences include formal tests for the result.

4.2.4 Save a report

The program should save reports on obtained solutions. Recommended formats are plain text, HTML, LaTeX (with compilation to PDF) or PDF.

A report must contain:

problem description;

steps of the algorithm;

solution;

graphical or tabular interpretation of results.

4.2.5 Get help

User should be able to find the information sufficient for use the application. This information can be provided in arbitrary form. Recommended formats are HTML and PDF. It is allowed to write the help in reStuctured Text or LaTeX and compile them to HTML or PDF.

5 COURSE WORK DEFENSE

5.1 General procedure of course work defense

Course work can be defended by students that fully completed their work. It is evidenced by the supervisor's review attached to report on the course work. The report must comply with the requirements of NTU "KhPI". The main conclusions of the supervisor (course work is done according to the task; course work is made without assistance and so on) should be reflected in his review.

Student must submit presentation materials and the developed application software.

Course work is defended using a computer. Student must put presentation materials and application software on the computer in advance. Course work defense begins with student's report, then he must answer the questions on the subject of his work.

Course work defense is public. Everyone can attended it and ask questions.

5.2 Requirements for presentation materials

Presentation materials should be present in the course work report in form of charts, tables, diagrams, etc. If presentation materials are missing in the body of the report (e.g., due to the fact that these materials provide information from different chapters of the report), they should be listed in appendices.

This is an example of a list of presentation materials.

1 Research problem statement.

2 Classification of basic algorithms.

3 Description of the class of used algorithms.

4 Scheme of the implemented algorithms.

5 Use case diagram.

6 Software requirements.

7 Development environment.

8 Data input.

9 Results (graphical or tabular interpretation).

10 Results (verifying solution).

11 Conclusions.

Presentations can be made in paper or electronic form. Paper presentation materials are carried out on A4 paper in typewritten form (i.e. should be printed on printer). All inscriptions and figures should be clear and easy to read. The handwritten version of presentation materials is not allowed. Electronic presentation materials are performed using appropriate software (LibreOffice Impress, Microsoft Power Point, etc.) and displayed by a computer.

5.3 Requirements for the report

The purpose of a report is to declare goals of the course work, identify and describe the main stages of its implementation and results. Reports takes up to 5 minutes. During the report, the student must use presentation materials. After the report, the student demonstrates the developed software.

5.4 Requirements for software demonstrations

The main objective of software demonstration is to show the performance of the developed software, its main functionality, user experience, and so on.

During the demonstration the student must show how to work with software in different modes:

– data input and modification;

– solving the problem;

– demonstration and verification of the solution.

6 CRITERIA OF THE COURSE WORK

The mark for the course work is affected by the following factors.

1 Errors, failures, etc. in the application software during its demonstration in course work defense.

2 Low-quality report.

3 Low-quality presentation materials that do not reflect the full features of the object domain, the results obtained in the course work and so on.

Supervisor's review is taken into account to give a mark. The review should reflect the following data:

– actuality of the work;

– degree of autonomy of student performance;

– main results obtained in the work;

– marks in the national scale and ECTS.

7 TASK VARIANTS FOR COURSE WORK

7.1 Research of efficiency of quadratic and optimal sorting algorithms

Input: array of N integers, N = 10 for the test and N = 10 000 000 for demonstration.

Data source: arbitrary video file, data is processed as a sequence of 32-bit integers in a format Big Endian.

Requirements for the interface: the user should be able to see every step of any sorting algorithm and obtain statistical information: how many permutations of the elements had to be performed, how many milliseconds it took to sort, etc.

Used algorithms: insertion sort, bubble sort, merge sort, heapsort, and quicksort.

7.2 Implementation and analysis of random number generators

Input: length of generated sequence N and initial parameters needed by generator; N = 100 for test and N = 10 000 for a demonstration.

Requirements for the interface: the user should be able to view all the generated sequence to observe its graphical interpretation:

histogram of elements distribution;

distribution on plane (element pairs are treated as coordinates (x, y));

autocorrelation (the user sets the offset for the copied sequence);

The sequence must be validated according to any (on student's choice) of four tests from NIST package.

Used algorithms: linear congruential method, lagged Fibonacci generator, generator from programming language standard library, reading from files /dev/random, /dev/urandom, and arbitrary other file.

7.3 Visualization of basic graph search algorithms

Input: a graph of N nodes given as set of its M edges; N, M = 10 for the test and N = 256, M = 2048 for a demonstration.

Data source: arbitrary video file, data is processed as a sequence of pairs of single-byte numbers - the numbers of vertices connected by an edge.

Requirements for the interface: the user should be able to see the graph on a plane and move the vertices using mouse for the best appearance; the results of

the movement and the graph itself can be saved to a file in a format chosen by student; the user should be able to choose a vertex and start the search from it; status (white, gray, black) and time of arrival and leaving must be shown for every vertex.

Used algorithms: breadth-first search, depth-first search.

7.4 Build minimum spanning tree for transport routes

Input: a graph of N nodes, the set of its M edges; N, M = 10 for the test and N = 256, M = 2048 for a demonstration.

Data source: arbitrary video file, data is processed as a sequence of pairs of single-byte numbers that are numbers of vertices connected by an edge.

Requirements for the interface: the user should be able to see the graph on a plane and move the vertices using the mouse for the best appearance;the results of the movement and the graph itself can be saved to a file in a format chosen by student (can be based on plain text, XML , JSON); user should be able to see the process of building the spanning tree step by step and get the result.

Used algorithms: algorithms of Kruskal and Prim.

7.5 Determination of software build order based on dependencies between components

Input: directed acyclic graph.

Data source: information about dependency software on user's computer, for example, the file /var/lib/dpkg/available.

Requirements for the interface: the user should be able to see the graph on a plane and move the vertices using the mouse for a better appearance, the user should be able to see the process of constructing the order of program build and save the results to a file.

Used algorithms: topological sort.

7.6 Building convex hull on plane

Input: a set of N points given by their integer coordinates; N = 20 for tests and N = 256 for a demonstration.

Data source: arbitrary video file, data is processed as a sequence of pairs of single-byte numbers - the coordinates of the vertices.

Requirements for the interface: the user should be able to see the point in the plane, move them with mouse; the results of movement can be saved to a file in the format chosen by student; the user should be able to see the building process step by step; the user should be able to point with mouse an arbitrary point on plane, and the program determine whether it belongs to the polygon corresponding to the hull.

Used algorithms: Jarvis march, Graham scan.

7.7 Finding the shortest path in a two-dimensional maze

Input: a set of N walls given by two pairs of integer coordinates of their vertices; N = 20 for the test and N = 50 for the demonstration.

Data source: arbitrary video file, data is processed as a sequence of single-byte numbers - coordinates of the vertices.

Requirements for the interface: the user should be able to see the walls of the plane, move them with mouse, the user can choose two points and the program will build between them the shortest path that does not intersect any wall or tell that such a path does not exist; user must be able to see the stages of routing.

Used algorithms: Dijkstra's or Floyd-Warshall algorithm.

7.8 Implementing encrypted messaging

Input: a user-selected text file and parameters of the algorithm

Requirements for the interface: the user can run two copies of the program on different computers, so they exchanged data over a network, or save an encrypted message to a file with the ability to decrypt it.

Used algorithms: RSA cipher or cipher of Vigenère.

7.9 Approximate solution of the traveling salesman problem

Input: N cities on plane defined by their integer coordinates; N = 20 for the test and N = 256 for a demonstration.

Source: arbitrary video file, data is processed as a sequence of single-byte numbers - coordinates of the vertices.

Requirements for the interface: the user should be able to see the city in a plane, move them with the mouse, the user can see the stages of routing and get

traveling salesman problem results indicating the length of the route and its sequence on plane.

Used algorithms: approximate solution of the traveling salesman problem.

7.10 Clustering of living organisms on the basis of the degree of similarity of their DNA

Input: N samples of DNA nucleotides of length M for various living organisms; N, M = 20 for the test and N, M = 256 for a demonstration.

Source: arbitrary video file, data is processed as a sequence of pairs of bits, each of which encodes a single nucleotide (adenine, thymine, guanine, cytosine).

Requirements for the interface: the user should be able to choose a pair of DNA samples and get their most common subsequence and build a dendrogram (cladogram) for all considered organisms.

Used algorithms: finding the longest common subsequence, agglomerative or divisive hierarchical clustering algorithm.

7.11 Development an application for archiving data based on Huffman code

Input: a user-selected file, the length of the block.

Requirements for the interface: the user can perform data compression and decompress archives, view statistics of each character density, additionally specify the length of blocks for file compression, the program should also have a command interface that is similar to archivers gzip, bzip2 or lzma.

Used algorithms: Huffman algorithm.

THE LIST OF SOURCES

1 N. Wirth. Algorithms + Data Structures = Programs. Prentice-Hall, 1976.

2 A. V. Aho, J. E. Hopcroft, J. D. Ullman, Data Structures and Algorithms. Addison-Wesley, 1983.

3 T. H. Cormen; C. E. Leiserson, R. L. Rivest, C. Stein. Introduction to Algorithms (3rd ed.). MIT Press and McGraw-Hill, 2009.

4 Игошин В.И. Математическая логика и теория алгоритмов : учеб. пособие для студ. высш. учеб. заведений / В. И. Игошин. – 2-е изд., стер. – М. : Издательский центр «Академия», 2008. – 448 с.

5 Игошин В.И. Задачи и упражнения по математической логике и теории алгоритмов / В. И. Игошин. – 3-е изд., стер. – М. : Издательский центр «Академия», 2007. – 304 с.

6 Романовский И.В. Вычислительная математика и структура алгоритмов. – М.: МГУ. 2006. – 112 с.

APPENDIXES

Appendix A Schedule of the course work

Table A.1 – Schedule of the course work on «Algorithms and Data Structures»

#	Name of course work stage	Time
1	Studying the theoretical foundations of used mathematical methods. Problem statement.	27 september
2	Algorithms for the problem	15 october
3	Developing software structure	30 october
4	Software development. Software implementation of used mathematical methods	
5	Solving and analyzing the results	10 november
6	Writing course work report	17 december
7	Preparation of course work presentation	22 december
8	Course work defense	25 december

Appendix B Use case diagram example

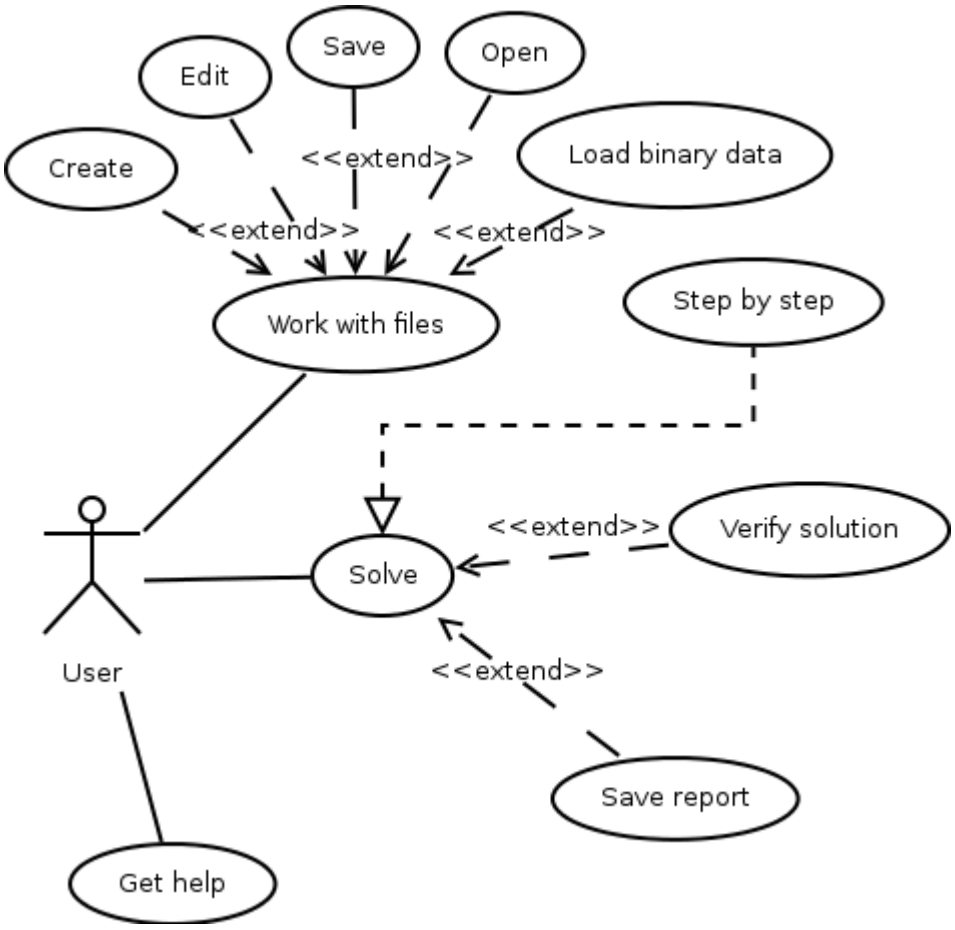

Figure B.1 – Use case diagram

Навчальне видання

Стратієнко Наталія Костянтинівна
Шматко Олександр Віталійович
Бородіна Інна Олександрівна

Guidance for course work on “Algorithms and Data Structures”
Методичні вказівки до виконання курсової роботи по курсу «Алгоритми та структури даних»

for students of direction 6.050103“Software Engineering”, specialty .05010302 “Software Engineering”
для студентів, які навчаються за напрямком 6.050103 «Програмна інженерія» спеціальності .05010301«Програмне забезпечення систем»
Англійською мовою

Відповідальний за випуск М.Д. Годлевський
Роботу до видання рекомендував О.В. Горілий

В авторській редакції

План 2016 р., поз. 80

Підписано до друку _____. Формат 60x84 1/16. Папір офсет.
Друк – ризографія. Гарнітура Times New Roman. Ум. друк. арк.
Наклад 50 прим. Зам № _____. Ціна договірна.

Видавничий центр НТУ «ХПІ». 61002, Харків, вул. Багалія, 21.
Свідоцтво про державну реєстрацію ДК № 3657 від 24.12.2009 р.

Друкарня НТУ «ХПІ». 61002, Харків, вул. Багалія, 21.